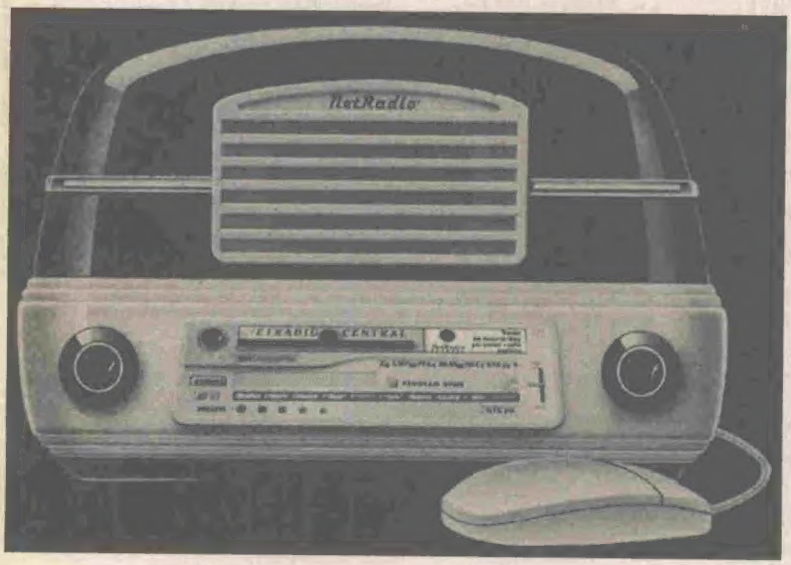
ซอฟต์แวร์

สุทธิศักดิ์ พงศ์ธนาพาณิช
(sutthisakp@yahoo.com)

เปิดโลกมัลติมีเดียด้วย Visual BASIC ตอน การสร้างโปรแกรม Audio CD Player ด้วยคอนโทรล ActiveX MMControl ตอนจบ



อธิบายถึงวิธี และโพรซีเดอร์เหตุการณ์ของคอนโทรล MMControl

จากบทความตอนที่แล้วผู้เขียนได้เขียนถึงเทคนิคการสร้างโปรแกรม Audio CD Player ซึ่งจะทำหน้าที่เล่นกลับเพลงจากแผ่น Audio CD โดยอาศัยความสามารถของคอนโทรล MMControl ซึ่งเป็นคอนโทรลที่สนับสนุนมาตรฐานตามสถาปัตยกรรม ActiveX ของ Visual BASIC 5.0 โดยได้กล่าวถึงคอนโทรล MMControl เกี่ยวกับรายละเอียด การใช้งานและคุณสมบัติทั้งหมดของคอนโทรล MMControl และสำหรับในบทความตอนนี้ ซึ่งเป็นตอนสุดท้ายของบทความนี้ ผู้เขียนก็จะได้อธิบายถึงวิธี และโพรซีเดอร์เหตุการณ์ของคอนโทรล MMControl และสำหรับในตอนท้ายของบทความตอนต่อไป ผู้เขียนก็จะแสดงวิธีการใช้งานคอนโทรล MMControl ด้วยโปรแกรมตัวอย่าง CDPlayer ซึ่งจะแสดงถึงเทคนิคในการเรียกใช้ความสามารถหลัก ๆ ของคอนโทรล MMControl ซึ่งผู้อ่านสามารถนำไปพัฒนาต่อเพื่อทำการเล่นกลับหรืออัดไฟล์เสียงหรือวิดีโอในรูปแบบของไฟล์ .wav, .mid หรือ .avi ได้ตามที่ต้องการ

สำหรับเนื้อหาในบางส่วนของบทความตอนนี้ (ผู้เขียนจะขออนุญาตไม่กล่าวถึงรายละเอียดทั้งหมดของโพรซีเดอร์เหตุการณ์ที่เกี่ยวข้องกับ OLE ซึ่งเป็นโพรซีเดอร์เหตุการณ์ที่มีชื่อขึ้นต้นด้วย OLE... โดยรายละเอียดทั้งหมดก็จะมีในหนังสือ) ผู้อ่านได้ทำการคัดลอกจากหนังสือ การใช้โปรแกรม Visual Basic 5.0 Professional Edition ซึ่งแต่งโดยตัวผู้เขียน และจัดพิมพ์และจำหน่ายโดย บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน) โดยคาดว่าจะออกวางจำหน่ายราว ๆ เดือนกรกฎาคม 2541 นี้ โดยที่เนื้อหาทั้งหมดของหนังสือเล่มนี้ ก็จะแสดงวิธีการใช้งานเอนไวรอนเมนท์และคอนโทรล ActiveX ทั้งหมดของ Visual Basic 5.0 Professional Edition สำหรับผู้อ่านที่สนใจก็สามารถหาอ่านได้จากร้านหนังสือชั้นนำทั่วไปครับ และสำหรับผู้อ่านท่านใดที่มีข้อสงสัยหรือต้องการสอบถามรายละเอียด ก็สามารถส่งอีเมล์มาได้ตามแอดเดรสข้างต้น

โพรซีเดอร์เหตุการณ์

ButtonClick	ButtonCompleted	ButtonGotFocus	ButtonLostFocus
Done	DragDrop	DragOver	GotFocus
LostFocus	OLECompleteDrag(5.0)	OLEDragDrop(5.0)	OLEDragOver(5.0)
OLEGiveFeedback(5.0)	OLESetData(5.0)	OLEStartDrag(5.0)	StatusUpdate

ButtonClick

เกิดขึ้นเมื่อผู้ใช้มีการคลิกเมาส์บนปุ่มคำสั่งของคอนโทรล MMControl

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_ButtonClick(Cancel As Integer)
```

Cancel หมายถึง ข้อมูลชนิดบูลีน ที่กำหนดการยกเลิกการปฏิบัติการของแต่ละปุ่มคำสั่งของคอนโทรล MMControl ที่ถูกคลิก ดังนี้

True หมายถึง ยกเลิกการปฏิบัติการของปุ่มคำสั่งของคอนโทรล MMControl ที่ถูกคลิก

False หมายถึง มีการปฏิบัติการของปุ่มคำสั่งของคอนโทรล MMControl ที่ถูกคลิก

ผู้อ่านสามารถกำหนดค่าให้กับพารามิเตอร์ Cancel ภายในรูปแบบโพรซีเดอร์ของเหตุการณ์ ButtonClick เพื่อให้มีการปฏิบัติการหรือยกเลิกการปฏิบัติการของปุ่มคำสั่งที่ต้องการได้

สำหรับแต่ละปุ่มคำสั่ง (โดยที่ชื่อของปุ่มคำสั่งจะปรากฏแทนที่ชื่อของโพรซีเดอร์เหตุการณ์ Button) จะมีการปฏิบัติการ โดยมีการเรียกใช้คำสั่ง MCI ดังตัวอย่างที่ 1

ปุ่มคำสั่ง	คำสั่ง MCI
Back	MCI_STEP
Step	MCI_STEP
Play	MCI_PLAY
Pause	MCI_PAUSE
Prev	MCI_SEEK
Next	MCI_SEEK
Stop	MCI_STOP
Record	MCI_RECORD
Eject	MCI_SET พร้อมกับพารามิเตอร์ MCI_SET_DOOR_OPEN

ตัวอย่างที่ 1

ButtonCompleted

เกิดขึ้นเมื่อคำสั่ง MCI มีการปฏิบัติการเสร็จสมบูรณ์

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_ButtonCompleted(Errorcode As Long)
```

Errorcode หมายถึง ข้อมูลชนิดเลขจำนวนเต็ม long ที่กำหนดรหัสข้อผิดพลาด ดังนี้

0 หมายถึง ไม่มีข้อผิดพลาดเกิดขึ้น เมื่อการปฏิบัติการเสร็จสมบูรณ์

<> 0 หมายถึง เกิดข้อผิดพลาดเกิดขึ้น หรือการปฏิบัติการไม่เสร็จสมบูรณ์

ButtonGotFocus

เกิดขึ้นเมื่อปุ่มคำสั่งของคอนโทรล MMControl ได้รับโฟกัส

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_ButtonGotFocus()
```

ButtonLostFocus

เกิดขึ้นเมื่อปุ่มคำสั่งของคอนโทรล MMControl สูญเสียสถานะของโฟกัส

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_ButtonLostFocus()
```

Done

เกิดขึ้นเมื่อคำสั่ง MCI ที่ถูกกำหนดให้คุณสมบัติ Notify มีค่าเท่ากับ True สิ้นสุดการปฏิบัติการ

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_Done(NotifyCode As Integer)
```

NotifyCode หมายถึง ข้อมูลชนิดเลขจำนวนเต็ม ที่กำหนดสถานะการปฏิบัติการของคำสั่ง MCI ดังนี้

ค่าตัวเลข	โหมดของดีไวซ์	รายละเอียด
1	mciNotifySuccessful	คำสั่งทำงานเสร็จสมบูรณ์
2	mciNotifySuperseded	คำสั่งถูกควบคุมหรือยกเลิกการทำงานโดยคำสั่งอื่น ๆ
4	mciNotifyAborted	คำสั่งถูกยกเลิกโดยผู้ใช้
8	mciNotifyFailure	คำสั่งทำงานล้มเหลว

StatusUpdate

เกิดขึ้นโดยอัตโนมัติทุก ๆ ช่วงระยะเวลาที่ถูกกำหนดโดยคุณสมบัติ UpdateInterval

รูปแบบโพรซีเดอร์เหตุการณ์

```
Private Sub MMControl_StatusUpdate()
```

โดยปกติผู้เขียนจะเขียนโค้ดลงในรูปแบบโพรซีเดอร์เหตุการณ์นี้ เพื่อที่การปรับปรุงความคืบหน้าในการทำงานและสถานะการทำงานของคำสั่ง MCI ในระหว่างการปฏิบัติการ

วิธี

Drag	Move	OLEDrag(5.0)	SetFocus
ShowWhatsThis	ZOrder		

Move

เคลื่อนย้ายตำแหน่งหรือปรับเปลี่ยนขนาดของคอนโทรล MMControl

รูปแบบการใช้งาน

```
object.Move left[, top, width, height]
```

left	หมายถึง	เลขจำนวนจริง single กำหนดตำแหน่งในแนวนอน (x) ของมุมบนซ้ายมือของคอนโทรล
top	หมายถึง	เลขจำนวนจริง single กำหนดตำแหน่งในแนวตั้ง (y) ของมุมบนซ้ายมือของคอนโทรล (optional)
width	หมายถึง	เลขจำนวนจริง single กำหนดความกว้างของคอนโทรล (optional)
height	หมายถึง	เลขจำนวนจริง single กำหนดความสูงของคอนโทรล (optional)

ซึ่งในการย้ายตำแหน่งการแสดงผลคอนโทรล MMControl ผู้อ่านสามารถกำหนดเฉพาะค่าของ left และ top เท่านั้นก็ได้ ส่วนค่าของ width และ height จะถูกใช้ในการกำหนดขนาดความกว้างและความสูงตามลำดับของคอนโทรล สำหรับหน่วยที่ใช้วัดขนาดของคอนโทรล MMControl จะถูกกำหนดโดยคุณสมบัติ ScaleMode

OLEDrag

กำหนดการเริ่มต้นการปฏิบัติการ DnD

รูปแบบการใช้วิธี

```
object.OLEDrag
```

ถ้าหากผู้อ่านเรียกวิธี OLEDrag เพื่อกำหนดการเริ่มต้นการปฏิบัติการ DnD ของ OLE ก็จะทำให้เกิดเหตุการณ์ OLEStartDrag ทันที

SetFocus

กำหนดให้คอนโทรลได้รับโฟกัส หรือเป็นคอนโทรลที่แอคทีฟ
รูปแบบการใช้งาน

```
object.SetFocus
```

สำหรับวิธีนี้ผู้อ่านจะสามารถใช้ได้กับคอนโทรลที่คุณสมบัติ Visible และ Enabled มีค่าเท่ากับ True เท่านั้น

ShowWhatsThis

กำหนดให้มีการแสดงผลข้อความหัวข้อที่ถูกกำหนดโดยใช้ What's This popup ซึ่งเป็นคุณสมบัติที่สนับสนุนโดย help ของวินโดวส์ 95
รูปแบบการใช้วิธี

```
object.ShowWhatsThis
```

ในการกำหนดหัวข้อในไฟล์ help ที่จะแสดงผลด้วยวิธีนี้ ผู้อ่านต้องกำหนดหมายเลข ID ของหัวข้อที่ต้องการให้กับคุณสมบัติ WhatsThisHelpID ของคอนโทรล

ZOrder

กำหนดลำดับการแสดงผลของคอนโทรลในแนวแกน Z (ทิศทางพุ่งออกจากระนาบจอภาพ)
รูปแบบการใช้งาน

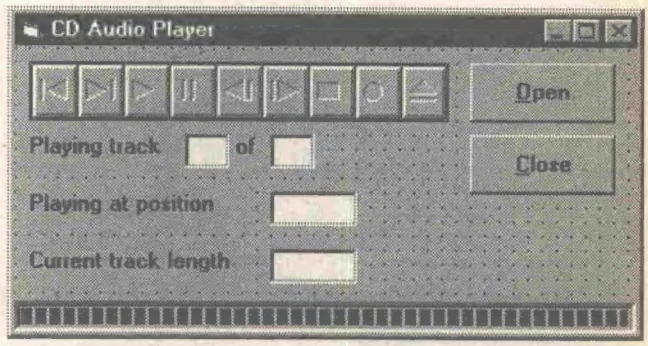
```
object.ZOrder position
```

position หมายถึง ลำดับการวางคอนโทรลลงในฟอร์ม โดยที่ถ้าหาก position มีค่าเท่ากับ 0 หรือมีการเรียกใช้วิธีนี้โดยไม่กำหนดค่าของ position คอนโทรลนั้นก็จะถูกวางเอาไว้บนสุด และในทางกลับกันถ้าหากฟอร์มใดที่มีค่าของ position มากที่สุดก็จะถูกวางไว้ล่างสุด ซึ่งในการแสดงผลก็จะถูกซ้อนทับโดยคอนโทรลที่มีค่า position น้อยกว่า

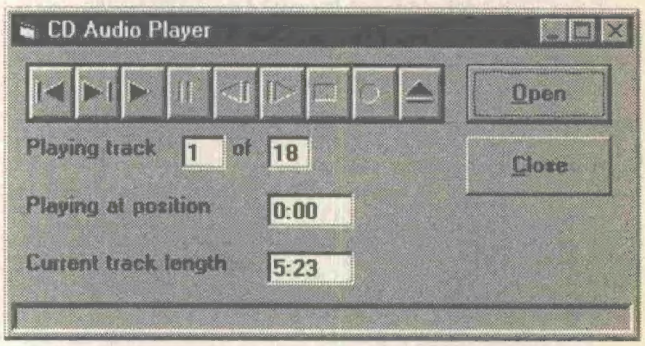
ในขณะออกแบบผู้อ่านสามารถกำหนดลำดับการซ้อนทับ (Z-order) ของคอนโทรลภายในแต่ละฟอร์ม ได้โดยการใช้คำสั่ง Bring To Front หรือ Send To Back ของเมนู Edit

โปรเจ็กต์ตัวอย่าง CDPlayer.vbp

โปรเจ็กต์ตัวอย่าง CDPlayer.vbp เป็นโปรเจ็กต์ง่าย ๆ ที่ประกอบด้วยฟอร์มเพียง 1 ฟอร์ม โดยในขณะออกแบบจะปรากฏฟอร์ม CDPlayer ดังในรูปที่ 1



รูปที่ 1 แสดงฟอร์ม CDPlayer ในขณะออกแบบ



รูปที่ 2 แสดงฟอร์ม CDPlayer ในขณะทำงาน

และเมื่อผู้อ่านรันโปรแกรมก็จะปรากฏฟอร์มดังในรูปที่ 2 โดยก่อนที่ผู้อ่านจะเล่น CD Audio ผู้อ่านก็ต้องคลิกที่ปุ่มคำสั่ง Open เสียก่อนเพื่อที่จะทำการเปิดดีไวซ์ จากนั้นก็ให้คลิกที่ปุ่มต่าง ๆ ของคอนโทรล MMControl เพื่อที่จะเล่นกลับ หยุด หรือ เปิด-ปิดถาดใส่ CD เป็นต้น ซึ่งความสามารถของปุ่มต่าง ๆ ของคอนโทรล MMControl ผู้อ่านสามารถอ่านได้จากบทความตอนที่ผ่านมาครับ

การทำงานของโปรเจ็กต์ตัวอย่าง CDPlayer.vbp

เมื่อมีการรันโปรแกรม โพรซีเดอร์ Form_Activate ก็จะถูกเรียกเมื่อฟอร์มถูกแสดงผล ซึ่งผู้เขียนได้เขียนโค้ดสำหรับการกำหนดชื่อของดีไวซ์ให้กับคอนโทรล MMControl1 ซึ่งในที่นี้หมายถึง CD Audio "CDAudio" โดยผ่านทางคุณสมบัติ DeviceType จากนั้นก็กำหนดให้คอนโทรล MMControl1 ทำการเรียกโพรซีเดอร์เหตุการณ์ StatusUpdate ทุก ๆ 1 วิธี โดยการกำหนดค่า 1000 ให้กับคุณสมบัติ UpdateInterval ดังนี้

```
Private Sub Form_Activate()
  MMControl1.DeviceType = "CDAudio"
  MMControl1.UpdateInterval = 1000
  MMControl1.TimeFormat = mciFormatMilliseconds
  MMControl1.Refresh
End Sub
```

เปิดโลกมัลติมีเดียด้วย Visual BASIC
ตอน การสร้างโปรแกรม Audio CD Player ด้วยคอนโทรล ActiveX MMControl ตอนจบ

จากนั้นเมื่อผู้อ่านคลิกที่ปุ่มคำสั่ง Open โพรซีเดอร์เหตุการณ์ Command1_Click ก็จะถูกเรียก ซึ่งผู้เขียนได้กำหนดให้มีการเปิดดีไวซ์ โดยกำหนดคำสั่ง "Open" ให้กับคุณสมบัติ Command และกำหนดให้ดีไวซ์รายงานตำแหน่งปัจจุบันและขนาดความยาวของเพลงในหน่วยของมิลลิวินาที ดังนี้

```
Private Sub Command1_Click()
   MMControl1.Command = "Open"
   MMControl1.TimeFormat = mciFormatMilliseconds
   MMControl1.Refresh
End Sub
```

และถ้าหากผู้อ่านต้องการปิดดีไวซ์ ก็ให้คลิกที่ปุ่มคำสั่ง Close โพรซีเดอร์เหตุการณ์ Command2_Click ก็จะถูกเรียก ซึ่งผู้เขียนได้กำหนดคำสั่ง "Stop" และ "Close" ให้กับคุณสมบัติ Command เพื่อให้ดีไวซ์หยุดการเล่นกลับก่อนที่จะปิดดีไวซ์ ดังนี้

```
Private Sub Command2_Click()
   MMControl1.Command = "Stop"
   MMControl1.Command = "Close"
   MMControl1.Refresh
   ProgressBar1.Value = 0
End Sub
```

สำหรับโค้ดส่วนที่สำคัญที่ทำการแสดงผลข้อมูลต่าง ๆ เกี่ยวกับ CD Audio และเพลงที่กำลังเล่นกลับในปัจจุบัน ก็จะถูกเขียนเอาไว้ในโพรซีเดอร์เหตุการณ์ MMControl1_StatusUpdate ซึ่งจะเป็นเทคนิคที่มีประโยชน์เป็นอย่างมาก ซึ่งเทคนิคที่สำคัญ ๆ มีดังต่อไปนี้

ตัวแปร CDTracksLength จัดเก็บขนาดความยาวของเพลงปัจจุบันในหน่วยของนาที

ตัวแปร CDTrackLength จัดเก็บขนาดความยาวของตำแหน่งที่กำลังเล่นกลับในขณะนั้น ๆ ของเพลงปัจจุบันในหน่วยของนาที

```
CDTracksLength = MMControl1.TrackLength / 60000
CDTrackLength = (MMControl1.Position -
MMControl1.TrackPosition) / 60000
```

คำนวณและแสดงผลขนาดความยาวของตำแหน่งที่กำลังเล่นกลับในขณะนั้น ๆ ของเพลงปัจจุบันในรูปแบบของ นาที:วินาที (mm:ss) ดังนี้

```
Text3 = Int(CDTrackLength)
& ":" & _Format((CDTrackLength -
Int(CDTrackLength)) * 60, "00")
```

คำนวณและแสดงผลขนาดความยาวของเพลงปัจจุบันในรูปแบบของ นาที:วินาที (mm:ss) ดังนี้

```
Text4 = Int(CDTracksLength)
& ":" & _Format((CDTracksLength -
Int(CDTracksLength)) * 60, "00")
```

สำหรับโค้ดต่อไปนี้ก็จะเป็นการอัพเดตคอนโทรล ProgressBar1 ทุก ๆ 1 วินาที โดยจะทำการอัพเดตก็ต่อเมื่อคุณสมบัติ MMControl1.TrackLength มีค่าไม่เท่ากับ 0 เท่านั้น และโค้ดที่สำคัญอีกประการ ก็คือเทคนิคการทำให้ดีไวซ์ทำการอัพเดตข้อมูลของเพลง ทุกครั้งที่มีการเริ่มต้นเล่นเพลงใหม่ถัดไป โดยการทดสอบดูว่าขนาดความยาวของตำแหน่งที่กำลังเล่นกลับอยู่ในขณะนั้น มากกว่าหรือเท่ากับขนาดความยาวของเพลงปัจจุบันหรือไม่ โดยถ้าหากมากกว่าก็แสดงว่า ในขณะนั้นดีไวซ์กำลังเล่นเพลงถัดไป ก็ให้กำหนดคำสั่ง"Prev" ให้กับคุณสมบัติ Command เพื่อให้ดีไวซ์ทำการอัพเดตข้อมูลของเพลงถัดไป ดังนี้

```
If MMControl1.TrackLength Then
   ' Update progressive bar every second tiggered.
   ProgressBar1.Value =
   Abs((MMControl1.Position - & _
      MMControl1.TrackPosition) * 100 / & _
      MMControl1.TrackLength) Mod 100

   ' Detect end of track.
   ' If playing until end of track then
   ' activate MCI command to update next track's
   information.
   If (MMControl1.Position -
      MMControl1.TrackPosition) >= & _
      MMControl1.TrackLength Then
      MMControl1.Command = "Prev"
      MMControl1.Refresh
   End If
End If
```

ส่วนโค้ดที่เขียนลงในโพรซีเดอร์เหตุการณ์ MMControl1_StopClick ก็เพื่อให้ดีไวซ์หยุดเล่นกลับเพลงปัจจุบัน และกลับไปยังตำแหน่งเริ่มต้นของเพลงปัจจุบัน เพื่อให้การเล่นกลับครั้งต่อไปสามารถเริ่มต้นจากตำแหน่งแรกสุดของแทร็กได้อย่างถูกต้อง เพื่อความเข้าใจขอให้ทดลองลบทิ้งโค้ดทั้งหมดในโพรซีเดอร์เหตุการณ์ MMControl1_StopClick แล้วรันโปรแกรม ก็จะเห็นถึงความแตกต่างเมื่อคลิกที่ปุ่ม Stop ของคอนโทรล MMControl1 ดังต่อไปนี้

```
Private Sub MMControl1_StopClick(Cancel As Integer)
   MMControl1.Command = "Stop"
   MMControl1.Refresh
   MMControl1.Command = "Prev"
   MMControl1.Refresh
   ProgressBar1.Value = 0
End Sub
```

สำหรับเนื้อหาทั้งหมดในบทความตอนนี้ได้กล่าวถึงเฉพาะการเล่นกลับ CD Audio ดังนั้นถ้าหากผู้อ่านสนใจที่จะศึกษาถึงเทคนิคการสร้างโค้ดเพื่อเล่นกลับไฟล์ .wav หรือ .mid ผู้อ่านก็สามารถอ่านได้จากหนังสือ การใช้โปรแกรม Visual Basic 5.0 Professional Edition ครับ พบกันใหม่ในตอนต่อไปสวัสดีครับ

```
Type=Exe
Form=CDPlayer.frm
Object={C1A8AF28-1257-101B-8FB0-0020AF039CA3}#1.1#0; MCI32.OCX
Object={6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0; COMCTL32.OCX
Object={C932BA88-4374-101B-A56C-00AA003668DC}#1.1#0; MSMASK32.OCX
IconForm="Form1"
Startup="Form1"
HelpFile=""
Title="CDPlayer"
Command32=""
Name="CDPlayer"
HelpContextID="0"
CompatibleMode="0"
MajorVer=1
MinorVer=0
RevisionVer=0
AutoIncrementVer=0
ServerSupportFiles=0
VersionComments="Multimedia API Programming by VB/Win"
VersionCompanyName="Microcomputer Magazine SE-ED"
VersionFileDescription="CD-Audio Player Program"
VersionLegalCopyright="Sutthisak Phongthanapanich"
CompilationType=0
OptimizationType=0
FavorPentiumPro(tm)=0
CodeViewDebugInfo=0
NoAliasing=0
BoundsCheck=0
OverflowCheck=0
FlPointCheck=0
FDIVCheck=0
UnroundedFP=0
StartMode=0
Unattended=0
ThreadPerObject=0
MaxNumberOfThreads=1
```

ลิสติงที่ 1
ไฟล์โปรเจ็กต์ CDPlayer.vbp

```
VERSION 5.00
Object = "{C1A8AF28-1257-101B-8FB0-0020AF039CA3}#1.1#0"; "MCI32.OCX"
Object = "{6B7E6392-850A-101B-AFC0-4210102A8DA7}#1.2#0"; "COMCTL32.OCX"
Begin VB.Form Form1
   Caption         =   "CD Audio Player"
   ClientHeight    =   2370
   ClientLeft      =   1575
   ClientTop       =   1515
   ClientWidth     =   5235
   BeginProperty Font
      Name            =   "MS Sans Serif"
      Size            =   8.25
      Charset         =   222
      Weight          =   700
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   ForeColor       =   &H80000008&
   LinkTopic       =   "Form1"
   PaletteMode     =   1  'UseZOrder
   ScaleHeight     =   2370
   ScaleWidth      =   5235
   Begin VB.TextBox Text4
      Height          =   285
      Left            =   2160
      TabIndex        =   11
      Top             =   1680
      Width           =   735
   End
   Begin VB.TextBox Text3
      Height          =   285
      Left            =   2160
      TabIndex        =   10
      Top             =   1200
      Width           =   735
   End
```

→

ลิสติงที่ 2
ไฟล์ฟอร์ม CDPlayer.frm

ลิสติงที่ 2 (ต่อ)

```
Begin VB.CommandButton Command2
   Caption         =   "&Close"
   Height          =   495
   Left            =   3840
   TabIndex        =   5
   Top             =   720
   Width           =   1215
End
Begin VB.CommandButton Command1
   Caption         =   "&Open"
   Height          =   495
   Left            =   3840
   TabIndex        =   4
   Top             =   120
   Width           =   1215
End
Begin VB.TextBox Text2
   Height          =   285
   Left            =   2160
   TabIndex        =   3
   Top             =   720
   Width           =   375
End
Begin VB.TextBox Text1
   Height          =   285
   Left            =   1440
   TabIndex        =   2
   Top             =   720
   Width           =   375
End
Begin ComctlLib.ProgressBar ProgressBar1
   Align           =   2  'Align Bottom
   Height          =   255
   Left            =   0
   TabIndex        =   1
   Top             =   2115
   Width           =   5235
   _ExtentX        =   9234
   _ExtentY        =   450
   _Version        =   327680
   Appearance      =   1
   MouseIcon       =   "CDPlayer.frx":0000
End
Begin MCI.MMControl MMControl1
   Height          =   495
   Left            =   120
   TabIndex        =   0
   Top             =   120
   Width           =   3540
   _ExtentX        =   6244
   _ExtentY        =   873
   _Version        =   327680
   DeviceType      =   "CDAudio"
   FileName        =   ""
End
Begin VB.Label Label4
   Caption         =   "Current track length"
   Height          =   255
   Left            =   120
   TabIndex        =   9
   Top             =   1680
   Width           =   1815
End
Begin VB.Label Label3
   Caption         =   "Playing at position"
   Height          =   255
   Left            =   120
   TabIndex        =   8
   Top             =   1200
   Width           =   1815
End
Begin VB.Label Label2
   Caption         =   " of "
   Height          =   255
   Left            =   1800
   TabIndex        =   7
   Top             =   720
   Width           =   375
End
Begin VB.Label Label1
   Caption         =   "Playing track : "
   Height          =   255
   Left            =   120
   TabIndex        =   6
   Top             =   720
   Width           =   1215
End
Attribute VB_Name = "Form1"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = False
Option Explicit
'
' Project name  : CDPlayer.vbp
' Program Name  : CD Audio Player Program
' VB/Win Module : CDPlayer.frm
' DLL filename  : -
' Copyright (c)1997 by Sutthisak Phongthanapanich
'

Private Sub Command1_Click()
    MMControl1.Command = "Open"
    MMControl1.TimeFormat = mciFormatMilliseconds
    MMControl1.Refresh
End Sub

Private Sub Command2_Click()
    MMControl1.Command = "Stop"
    MMControl1.Command = "Close"
    MMControl1.Refresh
    ProgressBar1.Value = 0
End Sub

Private Sub Form_Activate()
    MMControl1.DeviceType = "CDAudio"
    MMControl1.UpdateInterval = 1000
    MMControl1.TimeFormat = mciFormatMilliseconds
    MMControl1.Refresh
End Sub

Private Sub MMControl1_BackCompleted(Errorcode As Long)
    ProgressBar1.Value = 0
End Sub

Private Sub MMControl1_EjectCompleted(Errorcode As Long)
    ProgressBar1.Value = 0
End Sub

Private Sub MMControl1_NextCompleted(Errorcode As Long)
    ProgressBar1.Value = 0
End Sub

Private Sub MMControl1_PlayCompleted(Errorcode As Long)
    ProgressBar1.Value = 0
End Sub

Private Sub MMControl1_StatusUpdate()
    Dim CDTracksLength As Single
    Dim CDTrackLength As Single

    ' Convert millisecond to minute.
    CDTracksLength = MMControl1.TrackLength / 60000
    CDTrackLength = (MMControl1.Position -
MMControl1.TrackPosition) / 60000

    Text1 = MMControl1.Track
```

→

ลิสติงที่ 2 (ต่อ)

```
    Text2 = MMControl1.Tracks
    Text3 = Int(CDTrackLength) & ":" & _
            Format((CDTrackLength - Int(CDTrackLength)) * 60, "00")
    Text4 = Int(CDTracksLength) & ":" & _
            Format((CDTracksLength - Int(CDTracksLength)) * 60, "00")

    If MMControl1.TrackLength Then
        ' Update progressive bar every second tiggered.
        ProgressBar1.Value = & _
           Abs((MMControl1.Position - MMControl1.TrackPosition) & _
           * 100 / MMControl1.TrackLength) Mod 100
        ' Detect end of track.
        ' If playing until end of track then
        ' activate MCI command to update next track's information.
        If (MMControl1.Position - MMControl1.TrackPosition) >= & _
            MMControl1.TrackLength Then
            MMControl1.Command = "Prev"
            MMControl1.Refresh
        End If
    End If
End Sub

Private Sub MMControl1_StopClick(Cancel As Integer)
    ' Stop playing, then set to begin of current track.
    MMControl1.Command = "Stop"
    MMControl1.Refresh
    MMControl1.Command = "Prev"
    MMControl1.Refresh
    ProgressBar1.Value = 0
End Sub

Private Sub MMControl1_StopCompleted(Errorcode As Long)
    ProgressBar1.Value = 0
End Sub
```

เทคนิคประจำคอลัมน์

ถาม **รู้จักกับระบบวิดีโอของวินโดวส์ 95 ?**

ตอบ ระบบวิดีโอของวินโดวส์ 95 จะมีพื้นฐานอยู่บนระบบที่เรียกว่า audio-video interleave ซึ่งวินโดวส์ 95 จะจัดเก็บไฟล์วิดีโอไว้ โดยมีนามสกุล .avi โดยภายในไฟล์ดังกล่าวจะมีการจัดเก็บข้อมูลบิตแม็พและข้อมูลเสียง (wave) เป็นเซกเมนต์อย่างต่อเนื่อง

สำหรับข้อมูลที่ถูกจัดเก็บในไฟล์ .avi จะเป็นข้อมูลที่ถูกบีบอัดแล้ว ซึ่งอัลกอริธึมการบีบอัดข้อมูลสำหรับระบบวิดีโอของวินโดวส์ 95 นี้จะเป็นอัลกอริธึมแบบ lossy (ซึ่งมีหลายวิธี) ที่สามารถลดขนาดของข้อมูลได้ตั้งแต่ 10 เท่าขึ้นไป แต่ก็มีข้อเสียที่จะทำให้คุณภาพของวิดีโอและเสียงด้อยลงไปด้วย

ส่วนในขั้นตอนการเล่นกลับไฟล์ .avi วินโดวส์ 95 จะเริ่มต้นด้วยการตรวจสอบอัลกอริธึมที่ใช้ในการบีบอัดและคลายข้อมูล ที่เรียกกันว่า codec จากนั้นก็ทำการแยกข้อมูลบิตแม็พและข้อมูลเสียงออกจากกัน เพื่อทำการส่งข้อมูลบิตแม็พไปยังระบบแสดงผล และส่งข้อมูลเสียงไปยังการ์ดเสียง (sound card) พร้อม ๆ กันเพื่อจำลองการแสดงผลวิดีโอนั่นเอง